Laboratório 8
Programação concorrente em Python com threads e processos

Programação Concorrente (ICP-361)
Prof. Silvana Rossetto

[1]Instituto de Computação/CCMN/UFRJ - 2024-2

Introdução

O objetivo deste Laboratório é introduzir a programação concorrente em Python, mostrando o uso de threads e processos filhos. Para cada atividade, siga o roteiro proposto e responda às questões colocadas.

Atividade 1

Objetivo: Mostrar como criar threads em Python estendendo a classe Thread.

Roteiro: Abra o arquivo **helloPython.py** e entenda o que ele faz.

1. Execute o programa várias vezes, e observe os resultados impressos.

Atividade 2

Objetivo: Mostrar que é possivel compartilhar variaveis com threads, mas com processos não.

Roteiro: Abra o arquivo **incrementoThread.py** e siga o roteiro abaixo.

1. Leia e programa e compreenda como ele funciona.
2. Execute o programa várias vezes, e observe os resultados impressos. Os resultados foram os esperados?

Abra o arquivo **incrementoProcess.py** e siga o roteiro abaixo.

1. O que mudou em relação ao programa anterior?. Qual é o resultado esperado na saída?
2. Execute o programa várias vezes, e observe os resultados impressos. Os resultados foram os esperados?
3. Acompanhe a explanação da professora.

Atividade 3

Objetivo: Mostrar um exemplo de programa com uma thread que lê a entrada padrão.

Roteiro: Abra o arquivo **io.py** e siga o roteiro abaixo.

1. Leia e programa e compreenda como ele funciona.
2. Execute o programa e observe seus resultados. Qual o benefício da computação concorrente neste exemplo?
3. Acompanhe a explanação da professora.

Atividade 4

Objetivo: Mostrar um exemplo de programa *CPU-bound* que se beneficia da programação **multiprocesso** em Python.

Roteiro: Abra o arquivo **process.py** e siga o roteiro abaixo.

1. Leia e programa e compreenda como ele funciona.
2. Execute o programa e observe seus resultados. **Abra uma ferramenta de monitoramento do sistema e observe como os núcleos de execução da máquina estão sendo usados**.
3. Acompanhe a explanação da professora.

Agora vamos comparar essa execução com uma versão multithreading desse programa. Abra o arquivo **process_thread.py** e siga o roteiro abaixo.

1. Leia e programa e compreenda como ele funciona.
2. Execute o programa e observe seus resultados. **Abra uma ferramenta de monitoramento do sistema e observe como os núcleos de execução da máquina estão sendo usados**.
3. O que mudou em relação à execução da versão anterior?
4. Acompanhe a explanação da professora.

Atividade 5

Objetivo: Mostrar um exemplo básico de uso da programação multithreading/multiprocessos no desenvolvimento de aplicações distribuídas.

Roteiro:

- **Parte 1:** Introduz o conceito de servidor iterativo (*cli.py, srv.py*), apresenta questões referentes à API de comunicação por troca de mensagens e a implementação de aplicações distribuídas básicas e, por fim, introduz o uso da chamada de sistema "select" (*elegante-srv.py*) para permitir multiplexação de I/O (permitindo a interação do administrador com o servidor pela linha de comando).
- **Parte 2:** Mostra um exemplo de implementação em Python de um servidor iterativo que usa a chamada de sistema "select" para multiplexar o atendimento aos clientes (*multiplex-srv.py*).
- **Parte 3:** Mostra exemplos de implementação em Python de um servidor concorrente (*threads-srv.py, join-threads-srv.py e process-srv.py*).

Entrega: Não há entrega neste Lab. Tempo para o trabalho de implementação!